Subject: Re: How does IDL do ...
Posted by peter on Thu, 01 May 1997 07:00:00 GMT
View Forum Message <> Reply to Message

Eric Williams (ewilliams@wesleyan.edu) wrote:

: I am also curious as to how IDL does matrix calculations. A simple example:

: If you want to operate on an 2D array with FORTRAN you need to write
: nested loops to work through the rows and columns and work with each
: element.

: In IDL you apply a function or WHERE statement to a whole array in one command.
: I am wondering if IDL is still doing the nested loops anyway, and
: therefore not really any faster at doing the job?

Indeed, IDL is doing the nested loops internally (how could it not,
unless you have more than one processor). Its speed on such operations
is comparable to the explicit loop code in C or FORTRAN. If you
do the explicit loop in IDL, it will be very slow.

So, you don't get to go faster than FORTRAN. On the other hand, you
don't often go much slower. And your program will be working a week
before that other guys!

: Finally, I am also trying to sell IDL to students as one good tool learn
: not only for astronomy but to open up future job possibilities in other
: fields. I have mentioned the following fields:

: bio medical imaging

: Are these correct? Can anyone pass on a few more?

Yes for biomedical imaging.

More generally, matrix processing / data visualization environments like
IDL (and Matlab, Mathematica, name your favorite) are heavily used by
folks doing algorithm development and writing of in-house processing
applications. Knowing about such tools and how to exploit them is a
Good Thing. You can try out methods of data processing much more
rapidly in such an environment.

Peter

Subject: Re: How does IDL do ...
Posted by Liam Gumley on Fri, 02 May 1997 07:00:00 GMT

Peter Webb wrote:
[stuff deleted]
> So, you don't get to go faster than FORTRAN. On the other hand, you
> don't often go much slower. And your program will be working a week
> before that other guys!

Absolutely - don't try and sell IDL on the speed of execution - just tell people it's as fast as FORTRAN. Sell it on the basis of much faster development time (once you're up the learning curve a little bit).

A few other things about IDL that I really like:

(1) It's very easy to look at graphical output while you're developing a numerical analysis program. Often a plot of an array you are working on is much more informative than printing it as text.

(2) As long as you follow a few simple rules, you can port that whiz-bang graphical program from a Sun to an SGI to a laptop running Linux (of course porting to a PC or MAC can be a little more complicated, especially if you use widgets). I see people here who have developed large FORTRAN applications on one Unix box that they won't even *attempt* to port to another architecture - they know it will mean several weeks of frustration. Also, sharing IDL applications with your colleagues as XDR SAVE files is very useful. They don't even have to buy IDL first (grab a copy via FTP) - you can get a lot done in 7 minute demo mode (it forces you to start writing programs), especially since Postscript output and HDF input/output work just fine in demo mode.

(3) A lot of the time, it's fun! I had pretty much written FORTRAN code exclusively since I started college in 1983. In 1993 I first started using IDL, and it took a couple of years before I was writing well documented, reusable IDL procedures and functions. Now, I only use FORTRAN if I absolutely *have* to - anything new that I do is in IDL. And I'd have to say that I have a lot more fun programming in IDL that I ever did in FORTRAN. Once you've got the hang of writing FORTRAN-like numerical programs in IDL, there's still a whole different world to discover when you start writing widget-based applications, which are not like anything you've created before (if you're a FORTRAN programmer like me).

Just my $0.02 worth.

Cheers,
Liam.

Subject: Re: How does IDL do ...
Posted by nhbkmich on Fri, 02 May 1997 07:00:00 GMT
View Forum Message <> Reply to Message

Eric Williams (ewilliams@wesleyan.edu) wrote:
: If you want to operate on an 2D array with FORTRAN you need to write
: nested loops to work through the rows and columns and work with each
: element.

This is not true for the current Fortran standard (Fortran 90). It offers
array manipulations and an array syntax, that are quite similar to that
of IDL, but not identical.
I am using both of them and had to convert IDL sources to Fortran and
vice versa several times, usually in order to run calculations on
machines, that don't have IDL installed.
Due to their similarities, these conversions were straightforward and
easily done in most cases.

--
Michael Steffens Institut f. Meteorologie u. Klimatologie
Tel.: +49-511-7624413 Universitaet Hannover
email: Michael.Steffens@mbox.muk.uni-hannover.de Herrenhaeuser Str. 2
steffens@muk.uni-hannover.de D-30419 Hannover

PGP fingerprint = FA BE 6C 1C F6 C3 EC 33 DD 42 6B 7F DE CF 84 B8

Subject: Re: How does IDL do ...
Posted by Thomas A. McGlynn on Mon, 05 May 1997 07:00:00 GMT
View Forum Message <> Reply to Message

Peter Webb wrote:
>
> Eric Williams (ewilliams@wesleyan.edu) wrote:
>
> : I am also curious as to how IDL does matrix calculations. A simple example:
>
> : If you want to operate on an 2D array with FORTRAN you need to write
> : nested loops to work through the rows and columns and work with each
> : element.
>
> : In IDL you apply a function or WHERE statement to a whole array in one command.
> : I am wondering if IDL is still doing the nested loops anyway, and
> : therefore not really any faster at doing the job?
>
> Indeed, IDL is doing the nested loops internally (how could it not,
> unless you have more than one processor).

Actually this is not really correct and if you assume it you may
fall into a trap that I've hit more than once. E.g.,
suppose you have something like:

```
x = intarr(6) + 1
y = intarr(6)
y(x) = y(x) + 1
```

If you are thinking of IDL as simply hiding loops you may expect
to get y(1)=6 from these statements. In fact you get y(1)=1. IDL acts as if
the vector operations are occuring in parallel, even on a non-parallel
machine. In terms of computing efficiency, I essentially agree
with the statements that follow, though I imagine that there are cases
where IDL's loop operations will be a little faster than Fortran's.

> Its speed on such operations
> is comparable to the explicit loop code in C or FORTRAN. If you
> do the explicit loop in IDL, it will be very slow.
>
> So, you don't get to go faster than FORTRAN. On the other hand, you
> don't often go much slower. And your program will be working a week
> before that other guys!
...
> Peter

Regards,
Tom McGlynn
tam@silk.gsfc.nasa.gov